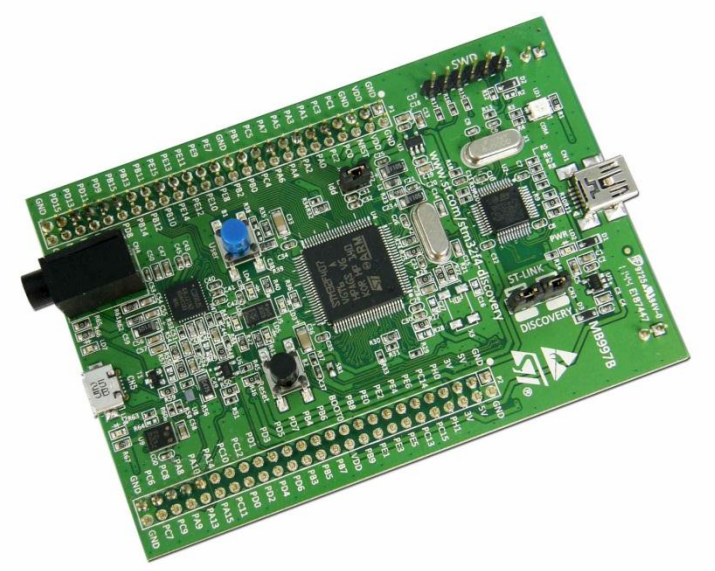
Politechnika Poznańska, Instytut Automatyki i Inżynierii Informatycznej

Podstawy technik mikroprocesorowych

Ćwiczenia laboratoryjne 4 – przerwania



Michał Fularz
2012-10-28

1. Wstęp teoretyczny.

Na zajęcia obowiązuje znajomość materiału zaprezentowanego na wykładzie.

2. Przebieg ćwiczenia.

a. Tworzenie projektu.

Utworzyć nowy projekt (lub skorzystać z programu z poprzednich zajęć). W zakładce ***Repositories*** dodać następujące moduły:

- MISC – funkcje obsługi kontrolera przerwań (dodaje pliki misc.h oraz misc.c do projektu),
- EXTI – funkcje odpowiedzialne za obsługę przerwań zewnętrznych (dodaje pliki stm32f4xx_exti.h oraz stm32f4xx_exti.c),
- SYSCFG – funkcje odpowiadające za podłączanie pinów wejść/wyjść do kontrolera przerwań zewnętrznych (dodaje pliki stm32f4xx_syscfg.h oraz stm32f4xx_syscfg.c).

W oknie edytora kodu otworzyć załączone pliki (aby je podświetlić w oknie ***Components*** wybrać odpowiedni moduł (MISC, EXTI lub SYSCFG)). Zwrócić uwagę na:

- Exported types (misc.h – NVIC_InitTypeDef, stm32f4xx_exti.h – EXTITrigger_TypeDef oraz EXTI_InitTypeDef),
- Exported constants (stm32f4xx_exti.h – EXTI_Line**X**),
- Exported functions.

W plikach misc.c oraz stm32f4xx_exti.c znajduje się krótki opis funkcjonalności danego modułu (na początku pliku, w komentarzu), a także definicje funkcji wraz z ich krótki opisem (indeksowanym przez IDE i wyświetlanym po umieszczeniu kursora na funkcji i naciśnięciu klawisza F2).

Do pliku z funkcją **main** programu dodać odpowiednie biblioteki (#include ”stm32f4xx_exti.h” oraz #include ”misc.h”).

Na początek programu dodać fragment kodu:

```
// ustawienie trybu pracy priorytetów przerwań
NVIC_PriorityGroupConfig(NVIC_PriorityGroup_1);
```

b. Konfiguracja przerwań – Timer/Counter.

W celu uaktywnienia przerwania należy stworzyć obiekt typu NVIC_InitTypeDef i wypełnić wszystkie pola, np.

```
NVIC_InitTypeDef NVIC_InitStructure;
```

```
// numer przerwania
NVIC_InitStructure.NVIC_IRQChannel = TIM3_IRQn;
// priorytet główny
NVIC_InitStructure.NVIC_IRQChannelPreemptionPriority = 0x00;
// subpriorytet
NVIC_InitStructure.NVIC_IRQChannelSubPriority = 0x00;
// uruchom dany kanał
NVIC_InitStructure.NVIC_IRQChannelCmd = ENABLE;
// zapisz wypełnioną strukturę do rejestrów
NVIC_Init(&NVIC_InitStructure);
```

Numerów przerwań należy szukać w pliku stm32f4xx.h jako wartości typu wyliczeniowego IRQn.

Po skonfigurowaniu kontrolera przerwań należy uaktywnić dane przerwanie w odpowiadającym module sprzętowym, w tym przypadku w TIM3 (timer numer 3).

```
// wyczyszczenie przerwania od timera 3 (wystąpiło przy konfiguracji
timera)
TIM_ClearITPendingBit(TIM3, TIM_IT_Update);
// zezwolenie na przerwania od przepełnienia dla timera 3
TIM_ITConfig(TIM3, TIM_IT_Update, ENABLE);
```

Powyższe funkcje można znaleźć w pliku stm32f4xx_tim.h (deklaracje) oraz stm32f4xx_tim.c (definicje). Dodatkowo w pliku stm32f4xx_tim.h są umieszczone wszystkie dostępne źródła przerwań dla timera (TIM_interrupt_sources).

Zadanie:

Skonfigurować timer 3 aby odmierzał czas 5 sekund, uruchomić dla niego przerwania (według procedury opisanej powyżej). W pętli głównej umieścić zamianę stanu diody co pewien czas (opóźnienie zrealizować za pomocą pustych operacji). Uruchomić program w trybie debug, zaobserwować działanie programu. Po upływie 5 sekund zatrzymać wykonywanie programu i zobaczyć w jakiej funkcji znajduje się program. Czy program działa poprawnie? Dlaczego dioda nie „mruga”?

c. Obsługa przerwań – Timer/Counter

W celu obsłużenia zgłoszonego przerwania należy napisać funkcję, która będzie wywołana w momencie wystąpienia danego przerwania.

Listę przerwań, adres funkcji ich obsługi oraz informacje na temat kontrolera przerwań można znaleźć w nocie katalogowej, w rozdziale Interrupts and events.

Szablon funkcja obsługi przerwania od przepełnienia dla timera 3:

```
void TIM3_IRQHandler(void)
{
	if(TIM_GetITStatus(TIM3, TIM_IT_Update) != RESET)
	{
		// miejsce na kod wywoływany w momencie wystąpienia przerwania

		// wyzerowanie flagi wyzwolonego przerwania
		TIM_ClearITPendingBit(TIM3, TIM_IT_Update);
```

```
    }
}
```

Zadanie:

Dodać funkcję obsługi przerwania do poprzedniego zadania. Czy teraz program działa poprawnie?

W funkcji obsługi przerwania zmieniać diodę LED, która aktualnie „mruga”.

d. Konfiguracja przerwań – przerwania zewnętrzne.

Przykład konfiguracji kontrolera przerwań do obsługi przerwania zewnętrznego z kanału **1**.

```
NVIC_InitTypeDef NVIC_InitStructure;
// numer przerwania
NVIC_InitStructure.NVIC_IRQChannel = EXTI1_IRQn;
// priorytet główny
NVIC_InitStructure.NVIC_IRQChannelPreemptionPriority = 0x00;
// subpriorytet
NVIC_InitStructure.NVIC_IRQChannelSubPriority = 0x00;
// uruchom dany kanał
NVIC_InitStructure.NVIC_IRQChannelCmd = ENABLE;
// zapisz wypełnioną strukturę do rejestrów
NVIC_Init(&NVIC_InitStructure);
```

W konfiguracji pinu procesora, oprócz standardowych funkcji konfiguracyjnych należy dodać konfigurację modułu przerwań zewnętrznych:

```
EXTI_InitTypeDef EXTI_InitStructure;
// wybór numeru aktualnie konfigurowanej linii przerwań
EXTI_InitStructure.EXTI_Line = GPIO_Pin_1;
// wybór trybu - przerwanie bądź zdarzenie
EXTI_InitStructure.EXTI_Mode = EXTI_Mode_Interrupt;
// wybór zbocza, na które zareaguje przerwanie
EXTI_InitStructure.EXTI_Trigger = EXTI_Trigger_Falling;
// uruchom daną linię przerwań
EXTI_InitStructure.EXTI_LineCmd = ENABLE;
// zapisz strukturę konfiguracyjną przerwań zewnętrznych do rejestrów
EXTI_Init(&EXTI_InitStructure);
```

Na koniec niezbędne jest połączenie wybranego pinu portu (w tym przypadku pin 1 z portu D) do modułu przerwań zewnętrznych:

```
// podłączenie danego pinu portu do kontrolera przerwań
SYSCFG_EXTILineConfig(GPIOD, EXTI_PinSource1);
```

e. Obsługa przerwań – przerwania zewnętrzne.

Analogicznie do obsługi przerwań od timera należy stworzyć funkcję, która będzie wywoływana w momencie wystąpienia przerwania. Przykład dla **1** kanału przerwań zewnętrznych.

```
void EXTI1_IRQHandler(void)
```

```
{
	if(EXTI_GetITStatus(EXTI_Line1) != RESET)
	{
		// miejsce na kod wywoływany w momencie wystąpienia przerwania

		// wyzerowanie flagi wyzwolonego przerwania
		EXTI_ClearITPendingBit(EXTI_Line1);
	}
}
```

Zadanie:

Korzystając z przykładów o konfiguracji i obsłudze przerwań zewnętrznych napisać program, w którym obsługa przycisku USER dostępnego na płytce jest realizowana przez przerwania zewnętrzne. **Uwaga** – sprawdzić w dokumencie (rozdział 4.11 – Extension connectors), do którego pinu podpięty jest przycisk.

W celu redukcji zjawiska drgań styku wykorzystać timer do ponownego testowania stanu pinu.

3. Zadania do samodzielnego wykonania realizacji.

- Z wykorzystaniem przerwań napisać program, który w regularnych odstępach czasu zmienia stan dostępnych diod LED (stworzyć własny, unikalny wzór). Naciśnięcie przycisku zmienia czas co jaki następuje przejście między stanami. Przytrzymanie przycisku przez czas dłuższy niż 3 sekundy powoduje zmianę trybu pracy (inny wzór). Wykorzystać przerwania od timera i przerwań zewnętrznych (np. wykrywać zbocze narastające i opadające).
- Korzystając z informacji przedstawionych na wykładzie oraz informacji zawartych w pliku misc.c dotyczących priorytetów (NVIC_IRQChannelPreemptionPriority oraz NVIC_IRQChannelSubPriority) napisać program, który zobrazuje działanie wywłaszczania przerwań. Przykład: stworzyć dwa timery, które odmierzają odpowiednio 5 oraz 10 sekund. W funkcjach obsługi przerwań umieścić nieskończoną pętlę, która zmienia stanu pinu od diody LED w pewnych odstępach czasu (opóźnienia za pomocą pustych pętli). Niech każda z funkcji obsługi timera kontroluje inną diodę. Modyfikując priorytety przerwań obserwować efekty wykonania programu.